Naključni sprehod bližine centralnosti

Urban Čretnik

Lenart Zavrtanik

Mentorja: prof. dr. Sergio Cabello, asist. dr. Janoš Vidali

[1. Uvod 3](#_Toc58865643)

[2. Definicije 3](#_Toc58865644)

[Centralnost bližine 3](#_Toc58865645)

[Prehodna matrika M 3](#_Toc58865646)

[Srednji čas prvega prehoda 3](#_Toc58865647)

[Naključni sprehod bližine centralnosti 4](#_Toc58865648)

[3. Potek dela 4](#_Toc58865649)

[4. Delo 4](#_Toc58865650)

[Generiranje grafov 4](#_Toc58865651)

[Zapis definicij 5](#_Toc58865652)

[Generiranje naključnih sprehodov 5](#_Toc58865653)

[5. Rezultati 6](#_Toc58865654)

[6. Zaključek 8](#_Toc58865655)

## Uvod

Naključni sprehod bližine centralnosti je merilo centralnosti omrežja, ki opisuje povprečno hitrost, s katero procesi naključnega sprehajanja dosežejo vozlišče iz drugih vozlišč v omrežju. Podobno je bližini centralnosti s to razliko, da se oddaljenost meri s pričakovano dolžino naključnega sprehoda, namesto s najkrajšo potjo. V projektni nalogi bova skušala dokazati, da z generiranjem naključnih sprehodov res dobimo podobne rezultate, ki bi jih dobili z uporabo enačb zapisanih v razdelku definicije.

## Definicije

V teoriji grafov in analizi omrežja *kazalniki centralnosti* identificirajo najpomembnejše točke v grafu. Vloge vključujejo identifikacijo najbolj vplivnih oseb v družbenem omrežju, ključnih infrastrukturnih vozlišč v internetu ali urbanih omrežjih in razširjevalce bolezni.

### Centralnost bližine

V povezanem grafu je normalizirana *centralnost bližine* vozlišča povprečna dolžina najkrajše poti med vozliščem in vsemi drugimi vozlišči v grafu. Tako bolj kot je vozlišče *centralno*, bližje je vsem ostalim vozliščem.

kjer je razdalja med oglišči x in y. Ko pa govorimo o središču bližine, se ljudje običajno sklicujemo na njegovo normalizirano obliko, ki jo ponavadi daje prejšnja formula, pomnožena s N-1, kjer je N število vozlišč na grafu . Ta prilagoditev omogoča primerjave med vozlišči grafov različnih velikosti.

### Prehodna matrika M

Če imamo usmerjen ali neusmerjen graf z n vozlišči označenimi z ; in proces naključnega sprehoda na tem omrežju s prehodno matriko . -ti element matrike opisuje *verjetnost naključnega sprehajalca, ki je dosegel vozlišče , da nadaljuje direktno v vozlišče* . Te verjetnosti so definirane na naslednji način:

kjer je -ti element matrike z utežmi omrežja . Ko med dvema vozliščema ni roba, je ustrezni element v matriki enak 0.

### Srednji čas prvega prehoda

*Srednji čas prvega prehoda od vozlišča do vozlišča*  je pričakovano število korakov, da proces prvič doseže vozlišče iz vozlišča :

kjer označuje verjetnost, da prvič potrebujemo natančno korakov, da dosežemo iz . Za izračun teh verjetnosti, da prvič dosežemo vozlišče v korakih, je koristno, da ciljno vozlišče obravnavamo kot absorpcijsko in da uvedemo transformacijo tako, da izbrišemo njegovo -to vrstico in stolpec in jo označimo z . Ker verjetnost procesa, ki se začne pri in je v po korakih, preprosto podaja -ti element , lahko označimo kot

Če to vstavimo v izraz za srednji čas prvega prehoda, dobimo

Z uporabo formule za seštevanje geometrijske vrste dobimo

Kjer je n-1 dimenzionalna identična matrika.

### Naključni sprehod bližine centralnosti

*Naključni sprehod bližine centralnosti* je merilo centralnosti omrežja, ki opisuje povprečno hitrost, s katero procesi naključnega sprehajanja dosežejo vozlišče iz drugih vozlišč v omrežju. Podobno je bližini centralnosti s to razliko, da se oddaljenost meri s pričakovano dolžino naključnega sprehoda, namesto s najkrajšo potjo.

Naključni sprehod bližine centralnosti vozlišča je inverzna povprečju srednjega časa prvega prehoda do tega vozlišča:

## Potek dela

Pogledala si bova različne tipe grafov: 2- dimenzionalne mreže, 3-dimenzionalne mreže, cikle in binarna drevesa. Pogledala si bova različne 2-dimenzionalnih mrež: 1x, 2x, x.

Naslednji postopek bova izvedla na različnih omrežjih:

1. Na vsakem omrežju bova prvo poiskala centralno vozlišče.
2. V centralnem vozlišču omrežja bova izračunala naključni sprehod bližine centralnosti po definiciji.
3. V centralnem vozlišču omrežja bova izračunala naključni sprehod bližine centralnosti z generiranjem naključnih sprehodov.
4. Rezultate iz 2. in 3. točke bova primerjala med sabo

Podatke bova primerjala s teoretičnimi trditvami in jih poizkušala potrditi oziroma ovreči.

## Delo

### Generiranje grafov

Za programiranje sva si izbrala program Sage, zaradi lažjega generiranja grafov. Naredila sva 4 različne funkcije za generiranje spodnjih tipov grafov;

* 2D mreže
* 3D mreže
* Binarna drevesa
* Cikle

### Zapis definicij

Najina naloga je bila, da z preizkusi potrdiva zgoraj navedene definicije. Torej po tem ko sva generirala grafe, sva morala na njih preveriti definicije. Za vsak graf sva tako po definiciji izračunala njegovo centralno vozlišče in vrednost. Vrednost sva zanemarila, saj je nisva imela s čim primerjati. Za lažje operiranje z grafi sva vsakemu priredila svojo incidenčno matriko. Da sva prišla do centralnega vozlišča sva prvo generirala prehodno matriko . Prehodno matriko sva dobila iz incidenčne matrike, kateri sva vsak pozitivni element matrike delila z vsoto vseh elementov te vrstice. Torej matrika nam za vsako vozlišče pove kakšne so verjetnosti, da se bo sprehod nadaljeval v sosednja vozlišča. Iz matrike sva nato morala narediti matriko . Vendar pa, če bi strogo sledila prvi (zgoraj zapisani) definiciji bi morala vmes definirati še verjetnostno matriko . Zato raje uporabiva tretjo (zgoraj zapisano) definicijo za generiranje matrike . Midva sva v nalogi naredila samo funkciji brisanje() in dodajanje(). Prva matriki izbriše j-to vrstico, druga doda vrstico samih ničelnih elementov. S pomočjo teh dveh funkcij sva direktno poračunala matriko . Nato sva definirala funkcijo oddaljenost\_i(), ki vrne najkrajšo pot od i-tega vozlišča do vseh ostalih in to funkcijo zapeljala po vseh vozliščih, da sva našla centralno. Torej tisto, ki je od vseh oddaljeno najmanj. Funkcija vrne tudi oddaljenost tega vozlišča do vseh ostalih vendar ta podatek ni uporaben za nadaljnje primerjave. Sedaj, ko imava centralno vozlišče naju je zanimalo kakšna je vrednost naključnih sprehodov iz tega do vseh ostalih. To vrednost sva po definiciji izračunala s funkcijo nakljucni\_sprehod\_blizine\_centralnosti(). Pri tej funkciji uporabiva že prej generirano matriko . Za i-to vozlišče funkcija vrne vsoto i-te vrstice matrike . Tako sva po definiciji izračunala podatke, ki sva jih v nadaljevanju z preizkusi preverjala.

### Generiranje naključnih sprehodov

Prvo sva zapisala funkcijo za naključni sprehod med dvema vozliščema v grafu. Funkcija za argument sprejme incidenčno matriko grafa in indeksa teh dveh vozlišč. Funkcija vrne dolžino naključnega sprehoda med tema dvema vozliščema, tj. število korakov, ki je bilo potrebnih da smo prišli iz enega vozlišča v drugo. Smiselno je poudariti, da če za argument funkcije izberemo enaki vozlišči funkcija vrne dolžino naključnega sprehoda do prve vrnitve v vozlišče (torej vsaj 2).

Nato sva naredila funkcijo en\_nsbc(), ki za argumenta sprejme incidenčno matriko in indeks vozlišča; funkcija naredi en naključni sprehod iz vsakega vozlišča do tega določenega vozlišča. Funkcija vrne seznam sprehodov od tega določenega vozlišča do vseh ostalih.

Naslednja funkcija n\_nakljucnih\_sprehodov() sprejme incidenčno matriko, indeks vozlišča in število ponovitev (n). Funkcija Naredi n naključnih sprehodov iz vsakega vozlišča do podanega vozlišča; se pravi da n-krat pokliče prejšnjo funkcijo.

Zadnja funkcija pa nato sešteje vse sprehode do določenega vozlišča. Število vseh naključno generiranih sprehodov deli s to vsoto.

## Rezultati

Generirala sva 6 različnih grafov:

1. A1…2D mreža z 1x100 vozlišči
2. A2…2D mreža s 2x50 vozlišči
3. A3…2D mreža s 10x10 vozlišči
4. B…3D mreža s 5x5x4 vozlišči
5. C…binarno drevo s 6 stopnjami
6. D…cikel s 100 vozlišči.

Na vsakem grafu sva prvo izračunala centralno vozlišče:

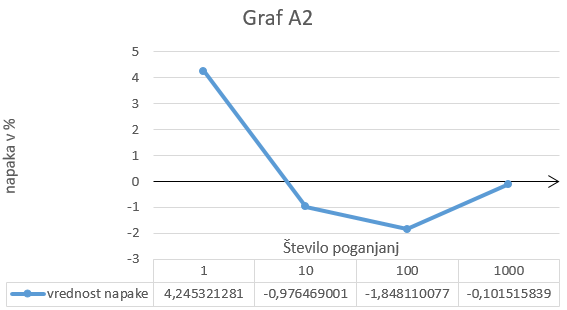
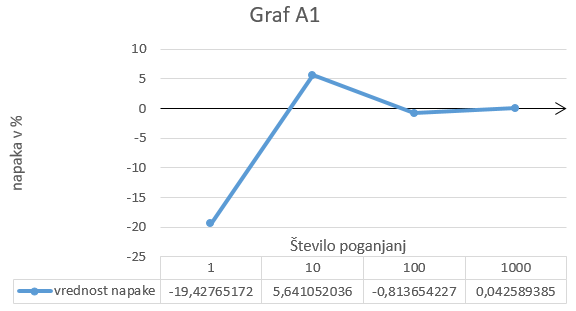
|  |  |  |
| --- | --- | --- |
| Graf | Indeks centralnega vozlišča | Vrednost |
| A1 | 49 | 0,0004 |
| A2 | 24 | 0,0007692307692307692 |
| A3 | 44 | 0,002 |
| B | 49 | 0,0029411764705882353 |
| C | 0 | 0,001557632398753894 |
| D | 0 | 0,0004 |

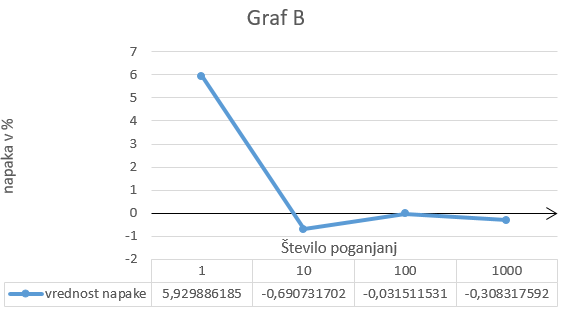
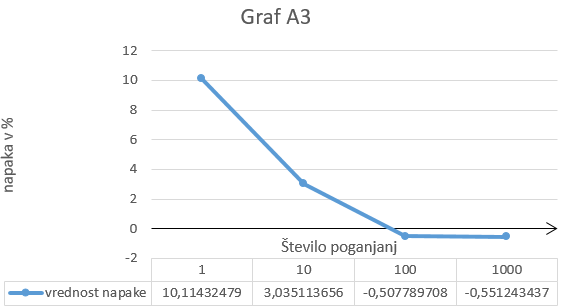
Opomba: centralnih vozlišč v določenem grafu je lahko več, midva sva si zabeležila samo tistega z najmanjšim indeksom.

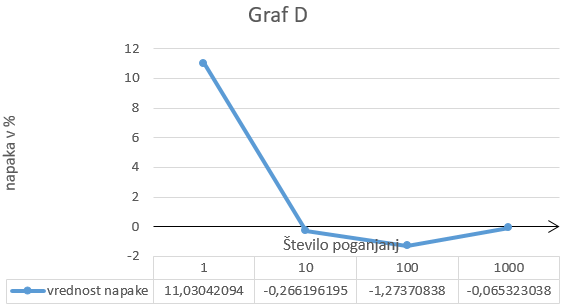
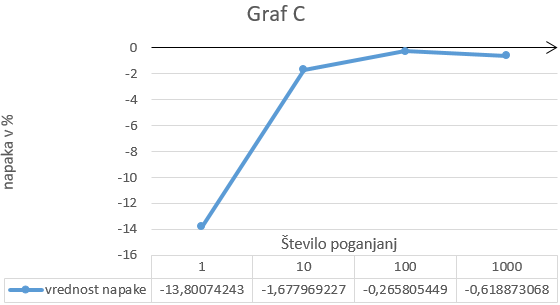
Opazimo, da se centralno vozlišče ujema z »vizualno najbolj centralnim«. V binarnem drevesu je centralno vozlišče pričakovano enako korenu grafa, v cikličnem grafu brez uteži pa so vsa vozlišča enako centralna.

Nato sva na vsakem od grafov v centralnem vozlišču po definiciji izračunala naključni sprehod bližine centralnosti.

Na vsakem od grafov sva v centralnem vozlišču generirala naključne sprehode in nato rezultate primerjala z dejanskim rezultatom izračunanem po definiciji. Na vsakem grafu sva naredila naslednje število naključnih sprehodov (poganjanj) z vsakega vozlišča: 1,10,100,1000. Rezultate sva prikazala v spodnjih grafih (napako sva merila v %):







Pričakovano pri vseh grafih dobimo največja odstopanja pri enem poganjanju. Že pri 10 poganjanjih pa dobimo občutno manjše napake. Vidimo da na različnih tipih grafov dobimo različne rezultate. Kljub sorazmerno majhen številu poganjanj (1000 ponovitev na 100 vozliščih) vsi rezultati (pri 1000 ponovitvah) odstopajo za manj kot 1%. Vsi rezultati pa zgleda, da konvergirajo k 0, torej se napaka z večanjem števila ponovitev (v povprečju) manjša. Smiselno bi bilo pričakovati, da bi dobili, če bi poslali število ponovitev proti neskončno, rezultat enak tistemu izračunanem po definiciji.

## Zaključek

Med izdelavo projekta sva definirala več funkcij, ki jih v sami nalogi nato nisva uporabila. Lahko bi npr. izračunala naključni sprehod bližine centralnosti v vseh vozliščih in ne samo v centralnem vozlišču. Rezultate bi lahko primerjala tudi na usmerjenih grafih oz. na grafih z utežmi. To bi lahko naredila z manjšimi popravki v kodi (popravek pri generiranju matrik prilagojenih grafom). Prav tako bi lahko naredila več ponovitev poizkusa oz. bi poizkus ponovila na več različnih grafih.

Največji problem zgoraj opisanih izboljšav je potreben čas. Samo generiranje matrik in nato računanje z njimi je časovno zelo potraten postopek. Z zgornjimi metodami bi torej verjetno dobila še boljše rezultate, kljub temu pa najini dobljeni rezultati kar dobro ustrezajo pričakovanim rezultatom.